5

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/755,449	01/13/2004	Vladimir Vasekin	550-500	1088

23117 7590 06/12/2007
NIXON & VANDERHYE, PC
901 NORTH GLEBE ROAD, 11TH FLOOR
ARLINGTON, VA 22203

EXAMINER	
GEIB, BENJAMIN P	
ART UNIT	**PAPER NUMBER**
2181	
MAIL DATE	**DELIVERY MODE**
06/12/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/755,449	VASEKIN, VLADIMIR	
	Examiner	Art Unit	
	Benjamin P. Geib	2181	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *23 March 2007*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-4,9-14,19-24,29 and 30* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-4,9-14,19-24,29 and 30* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-4, 9-14, 19-24, 29 and 30 have been examined.

2. It is hereby acknowledged that the following papers have been received and placed of record in the file: request for continued examination on 03/23/2007.

3. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on 03/23/2007 has been entered.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. Claims 1-4, 9-14, 19-24, 29, and 30 are rejected under 35 U.S.C. 103(a) as being unpatentable over Fraser, U.S. Patent No. 6,907,598 in view of Hennessy et al., "Computer Architecture: A Quantitative Approach" (Hereinafter Hennessy).

6. Referring to claim 1, Fraser has taught an apparatus for processing data, said apparatus comprising:

an instruction fetching circuit *[access module; Fig. 3, component 332]* operable to fetch program instructions from a sequence of memory locations *[column 11, line 65 – column 12, line 3]*;

an instruction decoder *[evaluate module; Fig. 3, component 334]* responsive to program instructions fetched by said instruction fetching circuit to control data processing operations specified by said program instructions *[column 12, lines 3-13]*; and

an execution circuit *[execute module; Fig. 3, component 338]* operable under control of said instruction decoder to execute said data processing operations *[column 12, lines 7-13]*;

a program counter register operable when said apparatus is executing said program instructions from said sequence of memory locations to store an address indicative of a memory location of a program instruction being executed within said program instructions from said sequence of memory locations *[The program counter stores a pointer indicating the next instruction to be executed; column 12, lines 5-7]*; and

a block counter register operable to store a block count value indicative of a location of a program instruction being executed within a block of two or more program instructions *[The count module stores the length parameter (i.e. block count value) of an echo instruction to determine the remaining instructions in the set of instructions (i.e. location of a program instruction being executed within the block); column 12, lines 32-40]*,

wherein said instruction decoder is responsive to an execute block instruction *[echo instruction]* to trigger fetching of a block of two or more program instructions *[set of instructions]* by said instruction fetching circuit and execution of said block of two or more program instructions by said execution circuit, said block of two or more instructions containing a number of program instructions specified by a block length field *[length parameter]* within said executed block instruction and being stored at a memory location specified by a location field *[displacement parameter]* within said execute block instruction *[When the evaluate module determines there is an echo instruction it triggers the fetching and execution of a set of instructions, whose number and location are indicated by the length and displacement parameters, respectively, of the echo instruction; column 12, lines 14-44]*; and

wherein when executing said block of two or more program instructions, said program counter register is configured to store an address indicative of a memory location of said execute block instruction rather than to store an address indicative of said memory location specified by said location field *[The program counter, during execution of an Echo instruction, will store an address of an instruction within the set of instructions other than the address of the first instruction in the set of instruction (i.e. an address indicative of the memory location specified by the location field)]* and said block counter register is configured to store a block count value indicative of said program instruction location of a program instruction being executed with said block of two or more program instructions *[the count of remaining instruction in the set of instruction]* corresponding to said execute block instruction *[column 12, lines 14-37]*,

Fraser has not explicitly taught an exception handling circuit operable upon occurrence of an exception during execution of said block of two or more instructions to store said block count value, and upon completion of handling of said exception, to restart execution of said block of two or more program instructions at a program instruction within said block of two or more instructions indicated by said block count value.

Hennessy has taught exception handling circuits that save a processor's execution state upon occurrence of an exception and restart execution at the program instruction indicated by the saved execution state upon completion of handling the exception *[Hennessy; pages A-42 - A-44]*.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to included an exception handling circuit that saves the processor's execution state, which in the instant case includes a block count value, upon occurrence of an exception and restarts execution at the program instruction indicated by the saved state, including the block count value, upon completion of handling the exception.

The suggestion/motivation for doing so would have been that the processor is allowed to properly resume execution of a program after an exception *[Hennessy; 1st paragraph of section label "Stopping and Restarting Execution" on page A-42]*.

7. Referring to claims 2, 12, and 22, taking claim 2 as exemplary, Fraser and Hennessy have taught the apparatus as claimed in claim 1, wherein after execution of said block of two or more program instructions a return is made to a program instruction

outside of said block of two or more program instructions *[Fraser; column 12, lines 32-44]*.

8. Referring to claims 3, 13, and 23, taking claim 3 as exemplary, Fraser and Hennessy have taught the apparatus as claimed in claim 1, wherein said return is to a program instruction immediately following said execute block instruction within said sequence of memory locations *[Fraser; column 12, lines 32-44]*.

9. Referring to claims 4, 14, and 24, taking claim 4 as exemplary, Fraser and Hennessy have taught the apparatus as claimed in claim 1, wherein said location field is an offset field *[Fraser; displacement parameter]* specifying said location of said block of two or more program instructions relative to a memory location of said execute block instruction *[Fraser; The displacement parameter of an echo instruction specifies the location of the set of instructions relative to the echo instruction; column 12, lines 20-30]*.

10. Referring to claims 9, 19, and 29, taking claim 9 as exemplary, Fraser and Hennessy have taught the apparatus as claimed in claim 1.

Fraser has not explicitly taught that the exception handling circuit is operable to store an address indicative of a memory location of said execute block instruction (i.e. the program counter) upon occurrence of an exception and to restore said address indicative of a memory location of said execute block instruction to said program counter register upon said completion of handling of said exception.

Hennessy has taught exception handling circuits that save a processor's execution state upon occurrence of an exception and restore the saved execution state upon completion of handling the exception *[Hennessy; pages A-42 - A-44].*

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the exception handling circuit to save the program counter (i.e. address indicative of a memory location of said execute block instruction), which is part of the processor's execution state, upon occurrence of an exception and restores the saved state, including the program counter, upon completion of handling the exception.

The suggestion/motivation for doing so would have been that the processor is allowed to properly resume execution of a program after an exception *[Hennessy; 1st paragraph of section label "Stopping and Restarting Execution" on page A-42].*

11. Referring to claims 10, 20, and 30, taking claim 10 as exemplary, Fraser and Hennessy have taught the apparatus as claimed in claim 2, wherein upon completion of execution of said block of two or more program instructions said instruction decoder is operable to return processing to a program instruction following said execute block instruction as indicated by said program counter register *[Fraser; Upon completion of the set of instructions the program counter is restored and the processing is restored to a program instruction following the echo instruction as indicated by the program counter; column 12, lines 37-44].*

12. Referring to claim 11, Fraser has taught a method for processing data, said method comprising the steps of:

fetching program instructions from a sequence of memory locations with an instruction fetching circuit *[access module; Fig. 3, component 332; column 11, line 65 – column 12, line 3]*;

controlling data processing operations specified by said program instructions with an instruction decoder *[evaluate module; Fig. 3, component 334; column 12, lines 3-13]*; and

executing said data processing operations with an execution circuit controlled by said instruction decoder *[execute module; Fig. 3, component 338; column 12, lines 7-13]*;

storing within a program counter register an address indicative of a memory location of a program instruction being executed within said sequence of program instructions *[The program counter stores a pointer indicating the next instruction to be executed; column 12, lines 5-7]*; and

storing within a block counter register a block count value indicative of a location of a program instruction being executed within a block of two or more program instructions *[The count module stores the length parameter (i.e. block count value) of an echo instruction to determine the remaining instructions in the set of instructions (i.e. location of a program instruction being executed within the block); column 12, lines 32-40]*,

wherein said instruction decoder is responsive to an execute block instruction *[echo instruction]* to trigger fetching of a block of two or more program instructions *[set of instructions]* by said instruction fetching circuit and execution of said block of two or

more program instructions by said execution circuit, said block of two or more instructions containing a number of program instructions specified by a block length field *[length parameter]* within said executed block instruction and being stored at a memory location specified by a location field *[displacement parameter]* within said execute block instruction *[When the evaluate module determines there is an echo instruction it triggers the fetching and execution of a set of instructions, whose number and location are indicated by the length and displacement parameters, respectively, of the echo instruction; column 12, lines 14-44]*,

wherein when executing said block of two or more program instructions, said program counter register stores an address indicative of a memory location of said execute block instruction rather than storing an address indicative of said memory location specified by said location field *[The program counter, during execution of an Echo instruction, will store an address of an instruction within the set of instructions other than the address of the first instruction in the set of instruction (i.e. an address indicative of the memory location specified by the location field)]* and said block counter register stores a block count value indicative of said program instruction location of a program instruction being executed within said block of two or more program instructions *[the count of remaining instruction in the set of instruction]* corresponding to said execute block instruction *[column 12, lines 14-37]*

Fraser has not explicitly taught upon occurrence of an exception during execution of said block of two or more instructions, storing said block count value and upon completion of handling of said exception, restarting execution of said block of two or

more program instructions at a program instruction within said block of two or more instructions indicated by said block count value.

Hennessy has taught saving a processor's execution state upon occurrence of an exception and restarting execution at the program instruction indicated by the saved execution state upon completion of handling the exception *[Hennessy; pages A-42 - A-44].*

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the method of Fraser to include saving the processor's execution state, which in the instant case includes a block count value, upon occurrence of an exception and restarting execution at the program instruction indicated by the saved state, including the block count value, upon completion of handling the exception.

The suggestion/motivation for doing so would have been that the processor is allowed to properly resume execution of a program after an exception *[Hennessy; 1st paragraph of section label "Stopping and Restarting Execution" on page A-42].*

13. Referring to claim 21, Fraser has taught a computer program product stored on a computer-readable storage medium including a computer program operable to control a data processing apparatus having

an instruction fetching circuit *[access module; Fig. 3, component 332]* operable to fetch program instructions from a sequence of memory locations *[column 11, line 65 – column 12, line 3],*

an instruction decoder *[evaluate module; Fig. 3, component 334]* responsive to program instructions fetched by said instruction fetching circuit to

control data processing operations specified by said program instructions *[column 12, lines 3-13]*, and

an execution circuit *[execute and Echo modules; Fig. 3, components 338 & 336]* operable under control of said instruction decoder to execute said data processing operations *[column 12, lines 7-13]* and to store within a program counter register an address indicative of a memory location of a program instruction being executed within said sequence of program instructions *[The program counter stores a pointer indicating the next instruction to be executed; column 12, lines 5-7]*;

wherein said computer program including one or more an execute block instructions *[echo instruction]* operable to trigger fetching of a block of two or more program instructions *[set of instructions]* by said instruction fetching circuit and execution of said block of two or more program instructions by said execution circuit and storing within a block counter register a block count value indicative of a location of a program instruction being executed within said block of two or more program instructions *[The count module stores the length parameter (i.e. block count value) of an echo instruction to determine the remaining instructions in the set of instructions (i.e. location of a program instruction being executed within the block); column 12, lines 32-40]*, said block of two or more instructions containing a number of program instructions specified by a block length field *[length parameter]* within said executed block instruction and being stored at a memory location specified by a location field *[displacement*

parameter] within said execute block instruction *[When the evaluate module determines there is an echo instruction it triggers the fetching and execution of a set of instructions, whose number and location are indicated by the length and displacement parameters, respectively, of the echo instruction; column 12, lines 14-44]*, and

wherein said one or more execute block instructions is operable to cause, during execution of said block of two or more program instructions, said program counter register to store an address indicative of a memory location of said execute block instruction rather than to store an address indicative of said memory location specified by said location field *[The program counter, during execution of an Echo instruction, will store an address of an instruction within the set of instructions other than the address of the first instruction in the set of instruction (i.e. an address indicative of the memory location specified by the location field)]* and to cause said block counter register to store a block count value indicative of said program instruction location of a program instruction being executed within said block of two or more program instructions *[the count of remaining instruction in the set of instruction]* corresponding to said execute block instruction *[column 12, lines 14-37]*; and

Fraser has not explicitly taught exception handling code, operable upon occurrence of an exception during execution of said block of two or more instructions, storing said block count value and upon completion of handling of said exception, restarting execution of said block of two or more program instructions at a program

instruction within said block of two or more instructions indicated by said block count value.

Hennessy has taught exception handling code, operable to save a processor's execution state upon occurrence of an exception and restart execution at the program instruction indicated by the saved execution state upon completion of handling the exception *[Hennessy; pages A-42 - A-44]*.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the system of Fraser to include exception handling code, operable to save the processor's execution state, which in the instant case includes a block count value, upon occurrence of an exception and restart execution at the program instruction indicated by the saved state, including the block count value, upon completion of handling the exception.

The suggestion/motivation for doing so would have been that the processor is allowed to properly resume execution of a program after an exception *[Hennessy; 1st paragraph of section label "Stopping and Restarting Execution" on page A-42]*.

Response to Arguments

6. Applicants arguments filed on March 23, 2007, have been fully considered but they are not found persuasive.

7. Applicant argues the novelty/rejection of claims 1-4, 9-14, 19-24, 29 and 30 on pages 10-14 of the remarks, in substance that:

> "Fraser fails to teach 'when executing said block of two or more program instructions, said program counter register is configured to store an address indicative of a memory location of said execute block instruction" (page 12)
>
> "[A]lthough the skilled person might have been aware based on the Hennessy text that the program counter value is saved and restored in the event of an exception, it would not have been obvious to that person to store a block count value – in addition to a program counter value – and to restart execution of the block of program instructions corresponding to the instruction within the block that was being executed when the exception occurred."

These arguments are not found persuasive for the following reasons:

The applicant argues that Fraser fails to teach "when executing said block of two or more program instructions, said program counter register is configured to store an address indicative of a memory location of said execute block instruction". As noted in the Advisory Action, the applicant appears to be reading the above-cited limitation too narrowly. In particular, the applicant appears to read the limitation as indicating that the program counter register stores the address in memory at which the execute block instruction itself is stored. However, this reading is not currently required by the claim. It is noted that the examiner and the applicant appear to be in agreement that the program counter of Fraser, during execution of an echo instruction, does not store the address in memory at which the echo instruction itself is stored, but instead stores an address of an instruction within the block of instructions indicated by the Echo instruction. However, the examiner and the applicant appear to be in disagreement about what is required by the above-cited limitation. The address stored in the program counter of Fraser during execution of an echo instruction is an address indicative of a memory location of an instruction that is executed as part of the echo instruction [Fraser; column 12, lines 14-37]. Since the stored address is indicative of a memory location of an instruction that is executed as part of the echo instruction the address is

indicative of a memory location of the echo instruction. If the applicant intends for the above-cited limitation to require that during execution of an execute block instruction the program counter stores an address in memory where the execute block instruction itself is stored, the examiner suggests the claim language "when executing said block of two or more program instructions, said program counter register is configured to store an address in memory where said execute block instruction is stored".

The applicant argues that it would not have been obvious to store a block count value in addition to a program counter value and to restart execution of the block of program instructions corresponding to the instruction within the block that was being executed when the exception occurred. Hennessy, which is relied upon for the above-mentioned feature, states that "the pipeline must be safely shut down and the state saved so that the instruction can be restarted in the correct state" [emphasis added] [Hennessy; 1st paragraph of section label "Stopping and Restarting Execution" on page A-42]. Therefore, Hennessy indicates that in order for execution to correctly resume state must be saved. Hennessy goes on to mention that typically this state comprises the "PC of the instruction at which to restart". In the system of Fraser, the program counter (PC) of the instruction at which restart (i.e. the next instruction to execute) is determined is not only determined by the current PC, but by the counter variable (i.e. block counter register) [Fraser; column 18, lines 16-30] and, therefore, the counter variable is part of the state needed in order to correctly resume execution. Further, Fraser has taught that the program counter and counter variable are stored upon execution of an echo instruction [Fraser; column 17, lines 47-56] and that storing this

information provides support for nested echo instructions [Fraser; column 19, lines 34-40] thereby further indicating that the is part of the state necessary to resume execution. Therefore, in view of Hennessy, it would have been obvious to store a block count value (i.e. counter variable) in addition to the program counter value and to restart execution of the block of program instructions corresponding to the instruction within the block that was being executed when the exception occurred.

Conclusion

14. The following is text cited from 37 CFR 1.111(c): In amending in reply to a rejection of claims in an application or patent under reexamination, the applicant or patent owner must clearly point out the patentable novelty which he or she thinks the claims present in view of the state of the art disclosed by the references cited or the objections made. The applicant or patent owner must also show how the amendments avoid such references or objections.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Benjamin P. Geib whose telephone number is (571) 272-8628. The examiner can normally be reached on Mon-Fri 8:30am-5:00pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Donald Sparks can be reached on (571) 272-4201. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Benjamin P Geib
Examiner
Art Unit 2181

DONALD SPARKS
SUPERVISORY PATENT EXAMINER